(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平11－143688

(43)公開日　平成11年(1999)５月28日

(51)Int.Cl.[6]	識別記号	ＦＩ	
Ｇ０６Ｆ　7/72		Ｇ０６Ｆ　7/72	
Ｇ０９Ｃ　1/00	６２０	Ｇ０９Ｃ　1/00	６２０Ｂ
			６２０Ｚ
	６５０		６５０Ａ

審査請求　未請求　請求項の数５　ＯＬ　（全 10 頁）

(21)出願番号　特願平9－304758

(22)出願日　平成９年(1997)11月６日

(71)出願人　000005223
富士通株式会社
神奈川県川崎市中原区上小田中４丁目１番１号

(72)発明者　岡田　壮一
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内

(72)発明者　鳥居　直哉
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内

(74)代理人　弁理士　河野　登夫

最終頁に続く

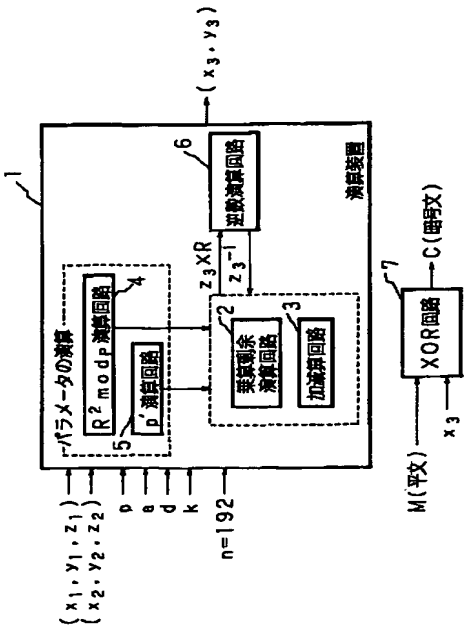
(54)【発明の名称】　演算装置並びにこれを利用したＲＳＡ暗号演算装置及び楕円暗号演算装置

(57)【要約】

【課題】　小さい回路規模で演算処理を行える、ＲＳＡ暗号演算と楕円暗号演算との両方が可能な装置を提供する。

【解決手段】　ＲＳＡ暗号演算装置に、加減算回路３と逆数演算回路６とを付加することにより、ＲＳＡ暗号処理と楕円暗号処理との両演算が可能である。乗算剰余演算回路２で、変数を格納するためにレジスタではなくワーキングメモリを使用して、回路規模を小型化を図る。

本発明の原理説明図



1

【特許請求の範囲】

【請求項１】　モンゴメリのアルゴリズムに応じた演算を含む演算処理を行う演算装置において、変数と途中の演算結果とを格納するワーキングメモリを設けたことを特徴とする演算装置。

【請求項２】　モンゴメリのアルゴリズムを利用して、ＲＳＡ暗号処理を行うＲＳＡ暗号演算装置において、変数と途中の演算結果とを格納するワーキングメモリを設けたことを特徴とするＲＳＡ暗号演算装置。

【請求項３】　モンゴメリのアルゴリズムを利用して、楕円暗号処理を行う楕円暗号演算装置において、変数と途中の演算結果とを格納するワーキングメモリを設けた乗算剰余演算回路と、モンゴメリ系での演算に必要なパラメータを演算するパラメータ演算回路と、加減算を行う加減算回路と、モンゴメリ系での逆数演算を行う逆数演算回路とを備えることを特徴とする楕円暗号演算装置。

【請求項４】　前記逆数演算回路は、モンゴメリの逆数アルゴリズムにおける２組４個の変数の加算処理を各組毎にそれぞれ並列的に行う２個の加算器を有する請求項３記載の楕円暗号演算装置。

【請求項５】　前記パラメータ演算回路及び／または前記逆数演算回路は、ビットのシフト処理を結線のつなぎ替えにて行うように構成した請求項３記載の楕円暗号演算装置。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、ＲＳＡ暗号の演算装置及び楕円暗号の演算装置、並びに、これらの演算装置における各種の演算回路に関する。

【０００２】

【従来の技術】高度情報化社会と呼ばれる現代社会では、コンピュータネットワークを基盤として、ビジネス上の重要な文書・画像情報が電子的な情報という形で伝送通信されて処理される。このような電子情報は、容易に複写が可能である、複写物とオリジナルとの区別が困難であるという性質があり、情報保全の問題が重要視されている。特に、無線を利用したネットワークにおいては、電子情報の傍受が容易なために、これを防止する対策が望まれている。このような対策の有効な手法として、人類の過去の歴史上で主として軍事，外交面に用いられてきた暗号技術が注目されており、種々のネットワークにも導入されつつある。

【０００３】暗号とは、情報の意味が当事者以外には理解できないように情報を変換することである。暗号において、誰でも理解できる元の文（平文）を第三者には意味がわからない文（暗号文）に変換することが暗号化であり、また、暗号文を平文に戻すことが復号化である。この暗号化の過程及び復号化の過程には、それぞれ暗号鍵及び復号鍵と呼ばれる秘密の情報が用いられる。

2

【０００４】このような鍵を用いる暗号化方式には、秘密鍵暗号系と公開鍵暗号系とが存在する。秘密鍵暗号系は、送信者と受信者とが同じ暗号鍵を持つことによって暗号通信を行う方式であり、送信者はある情報を秘密の暗号鍵に基づいて暗号化して受信者に送り、受信者はこの暗号鍵を用いて暗号を復号して元の情報を得る。これに対して、公開鍵暗号系では、公開されている受信者の公開鍵で送信者が情報を暗号化して送信し、受信者は自分の秘密鍵でその暗号を復号して元の情報を得る。

【０００５】公開鍵暗号系の中で最も有力な暗号系として、1977年にRivest，Shamir及びAdleman の３人によって発明されたＲＳＡ暗号がある。以下、このＲＳＡ暗号の原理について簡単に説明する。

【０００６】ＲＳＡ暗号では、暗号鍵（ｅ，Ｎ）、復号鍵（ｄ，Ｎ）、平文をＭ、暗号文をＣとした場合、暗号化Ｅと復号化Ｄのアルゴリズムは、べき乗剰余演算を用いて次のように表される。

Ｃ＝Ｅ（Ｍ）＝$M^e \bmod N$

Ｍ＝Ｄ（Ｃ）＝$C^d \bmod N$

但し、$e \cdot d \equiv 1 \bmod [LCM\{(p-1),(q-1)\}]$，

Ｎ＝ｐ・ｑ（ｐ，ｑは大きな素数）

ｅ，Ｎ：公開鍵

ｄ：秘密鍵

ＬＣＭ（Lowest Common Multiple）：最小公倍数

【０００７】上記式の場合に、それぞれｅ回の乗算と除算、ｄ回の乗算と除算を行わなければならず、通常、ｅ，ｄ，Ｍ，Ｎは 512ビット以上の大きな整数が用いられ、高速指数演算法を用いても、１回のＲＳＡ演算で平均 770回の乗算と剰余演算とを行わなければならない。よって、ＲＳＡ暗号演算を高速に行うためには、剰余演算の高速化が必要不可欠であり、特に剰余演算は、近似法，モンゴメリのアルゴリズム等、多くの高速化手法が提案されている。

【０００８】ＲＳＡ暗号に代表される公開鍵暗号系の多くで利用されるべき乗剰余アルゴリズムを高速に処理するためには、１回あたりの剰余アルゴリズムの高速化が要求される。このような観点に基づいて、本発明者等は、剰余演算の高速化を実現するべく、モンゴメリのアルゴリズムを用いた乗算剰余演算装置を提案している（特開平７－20778 号公報）。この乗算剰余演算装置では、モンゴメリ法による剰余演算における繰り返し処理の部分にテーブルを用いて、効率良く剰余演算処理を高速化し、剰余演算の処理速度を向上させる。

【０００９】ここで、モンゴメリのアルゴリズムについて簡単に説明する。モンゴメリのアルゴリズムは、剰余の法Ｎ（Ｎ＞１）と、剰余の法Ｎと互いに素である基数Ｒとを用いると、被剰余数ＴからＴR^{-1} modＮの計算が基数Ｒによる除算のみで行えることを利用して、Ｎによる除算を用いることなく剰余計算を行うアルゴリズムで

3

ある。ここで、N，N′，R，R^{-1}及びTは整数であり、被剰余数Tは$0 \leqq T < RN$、R^{-1}は剰余の法Nの上での基数Rの逆数であり、$RR^{-1} - NN' = 1$（$0 \leqq R^{-1} < N$，$0 \leqq N' < R$）の関係を満たす。

【００１０】更に、この基数Rに２のべき乗数を使用した場合、基数Rによる除算をシフト操作に置き換えることができるため、$T \rightarrow TR^{-1} \bmod N$の計算の高速処理が可能となる。具体的なアルゴリズムとして、$T \rightarrow TR^{-1} \bmod N$のアルゴリズムREDC（T）を以下に示す。但し、このアルゴリズムにおいて、（T＋mN）／Rは必ず割り切れる。

$T \rightarrow TR^{-1} \bmod N$のアルゴリズムREDC（T）

$$\begin{aligned} &REDC(REDC(T) \times (R^2 \bmod N)) \\ &= (TR^{-1} \bmod N)\ (R^2 \bmod N)\ R^{-1} \bmod N \\ &= TR^{-1} \times R^2 \times R^{-1} \bmod N \\ &= T \bmod N \end{aligned}$$

このようにして、簡単にT modNを求めることができる。

【００１２】

【発明が解決しようとする課題】特開平７－20778 号公報に示した乗算剰余演算装置では、モンゴメリのアルゴリズムでの変数はレジスタに格納しているが、ＲＳＡ演算にあっては、演算処理の高速化に加えてその演算装置の回路規模の小型化も重要な問題である。

【００１３】ところで、ＲＳＡ暗号は、公開鍵暗号の中でも、有力で実用的な暗号形式であると言われており、ＲＳＡ暗号用の演算チップの開発が積極的に進められている。しかし、近年、ＲＳＡ暗号の強度の低下について言及されるようになり、ＲＳＡ暗号に代わる公開鍵暗号として、楕円暗号が注目されている。

【００１４】楕円暗号の基本演算について、以下に記述する。有限体Ｆｐ（ｐはｐ＞３の素数）において、Ｆｐ上の楕円曲線Ｅ（Fp）は、

$y^2 = x^3 + ax + b$

但し、ａ，ｂは$0 \leqq a$，$b < p$なる整数であり、

$4a^3 + 27b^2 \neq 0 \pmod p$

で表され、楕円曲線上の点とは、Ｅ（Fp）上の点の組と無限遠点Ｏとのことを言う。

【００１５】この楕円曲線Ｅ（Fp）上の点は、以下の加算ルールで加算を行う。

（１）Ｏ＋Ｏ＝Ｏ

（２）（ｘ，ｙ）＋Ｏ＝（ｘ，ｙ）

（３）（ｘ，ｙ）＋（ｘ，－ｙ）＝Ｏ

（点（ｘ，ｙ）と点（ｘ，－ｙ）とは逆数の関係にある）

（４）逆数の関係にない２点（x_1，y_1），（x_2，y_2）に対して、加算結果を（x_3，y_3）とする。

①（x_1，y_1）≠（x_2，y_2）の場合

（x_1，y_1）＋（x_2，y_2）＝（x_3，y_3）

但し、$x_3 = \lambda^2 - x_1 - x_2$

4

$*m = (T \bmod R)\ N' \bmod R$

$t = (T + mN) / R$

もし、ｔ＜Ｎの場合にはｔになり、それ以外はｔ－Ｎ即ち、

$REDC(T) = t \quad (t < N)$

$REDC(T) = t - N \quad (t \geqq N)$

【００１１】１回のREDCでは、剰余T modNではなくT R^{-1} modNが求められるだけである。そこで、剰余T modNを求めるためには、以下に示すように、REDC（T）と予め求めておいたR^2 mod Ｎとの積で、再びREDCを行えばよい。

$y_3 = \lambda (x_1 - x_3) - y_1$

$\lambda = (y_1 - y_2) / (x_1 - x_2)$

②（x_1，y_1）＝（x_2，y_2）の場合

２（x_1，y_1）＝（x_3，y_3）

但し、$x_3 = \lambda^2 - 2x_1$

$y_3 = \lambda (x_1 - x_3) - y_1$

$\lambda = (3x_1^2 + a) / 2y_1$

【００１６】次に、楕円曲線上の点を高速に加算する手法を説明する。まず、楕円曲線上の点を三次元空間で表現する。即ち、（ｘ，ｙ）を（ｘ／ｚ，ｙ／ｚ）と対応させる（三次元投影法）。三次元で表した点の座標をモンゴメリ系に変換し、加算ルールに従って点の加算（各座標では、mod ｐ 上で加減乗算）を行う。最後に、ｚ座標の逆数$z^{-1} \bmod p$を求め、それをｘ座標及びｙ座標と乗算して、点の加算結果を得る。

【００１７】上述したような楕円暗号演算を行う場合、上記の加算ルール（４）のように、毎回、除算を行わなければならないし、三次元投影法にモンゴメリ法を適用して加算処理を行う手法においても、最後に、逆数を計算しなければならない。ＲＳＡ暗号演算を行う回路構成では、除算，逆数演算を行うことができず、ソフトウエアでこれらの演算処理を行うと、長時間要するという問題がある。例えば192ビットの逆数演算を10000 回行った場合、拡張ユークリッドでは約12秒、モンゴメリインバースアルゴリズムでは、約7.3 秒かかる（Pentium 133 ＭＨｚ）。

【００１８】本発明は斯かる事情に鑑みてなされたものであり、小さい回路規模で演算処理が可能であり、変数のビット長の拡張にも対応できる演算装置、及び、ＲＳＡ暗号演算装置を提供することを目的とする。

【００１９】本発明の他の目的は、ＲＳＡ暗号演算装置に、加減算回路と逆数演算回路とを付加することにより、楕円暗号演算を可能とする楕円暗号演算装置を提供することにある。

5

【００２０】本発明の更に他の目的は、小さい回路規模で演算時間が早い楕円暗号演算装置を提供することにある。

【００２１】

【課題を解決するための手段】請求項１に係る演算装置は、モンゴメリのアルゴリズムに応じた演算を含む演算処理を行う演算装置において、変数と途中の演算結果とを格納するワーキングメモリを設けたことを特徴とする。

【００２２】請求項２に係るＲＳＡ暗号演算装置は、モンゴメリのアルゴリズムを利用して、ＲＳＡ暗号処理を行うＲＳＡ暗号演算装置において、変数と途中の演算結果とを格納するワーキングメモリを設けたことを特徴とする。

【００２３】請求項３に係る楕円暗号演算装置は、モンゴメリのアルゴリズムを利用して、楕円暗号処理を行う楕円暗号演算装置において、変数と途中の演算結果とを格納するワーキングメモリを設けた乗算剰余演算回路と、モンゴメリ系での演算に必要なパラメータを演算するパラメータ演算回路と、加減算を行う加減算回路と、モンゴメリ系での逆数演算を行う逆数演算回路とを備えることを特徴とする。

【００２４】請求項４に係る楕円暗号演算装置は、請求項３において、前記逆数演算回路は、モンゴメリの逆数アルゴリズムにおける２組４個の変数の加算処理を各組毎にそれぞれ並列的に行う２個の加算器を有することを特徴とする。

【００２５】請求項５に係る楕円暗号演算装置は、請求項３において、前記パラメータ演算回路及び／または前記逆数演算回路は、ビットのシフト処理を結線のつなぎ替えにて行うように構成したことを特徴とする。

【００２６】図１は、本発明の原理説明図である。図において１は、ＲＳＡ暗号及び楕円暗号の演算を行う演算装置である。演算装置１は、モンゴメリ系での乗算を行う乗算剰余演算回路２と、加算・減算を行う加減算回路３と、モンゴメリ系での演算に必要なパラメータ$R^2 \bmod p$を計算する$R^2 \bmod p$演算回路４と、モンゴメリ系での演算に必要なパラメータp′を計算するp′演算回路５と、入力変数ｚの逆数z^{-1}を求める逆数演算回路６＊とを有する。また、７は排他的論理和を求めるＸＯＲ回路である。

6

【００２７】図１の構成は、ＲＳＡ暗号演算装置に加減算回路３及び逆数演算回路６を付加したものであって、ＲＳＡ暗号演算を行えるだけでなく楕円暗号演算も行える構成である。

【００２８】次に、本発明の動作原理について説明する。モンゴメリ系への変換処理は、乗算剰余演算回路２と$R^2 \bmod p$演算回路４とを用いて行う。与えられる開始点Ｐ（x_0/z_0，y_0/z_0）とすると、

$X_0 = \mathrm{REDC}(x_0 \times R^2 \bmod p)$
$= x_0 \times R^2 \times R^{-1} \bmod p$
$= x_0 \times R \bmod p$
$Y_0 = \mathrm{REDC}(y_0 \times R^2 \bmod p)$
$= y_0 \times R^2 \times R^{-1} \bmod p$
$= y_0 \times R \bmod p$
$Z_0 = \mathrm{REDC}(z_0 \times R^2 \bmod p)$
$= z_0 \times R^2 \times R^{-1} \bmod p$
$= z_0 \times R \bmod p$

となる。但し、この変換は、開始点だけについて行う。

【００２９】モンゴメリ系への変換後は、加算ルールに従って加算を行う。その結果、以下の式に基づく演算をmod p の上で行うことになる。

【００３０】点（x_1，y_1，z_1），（x_2，y_2，z_2）の演算結果を点（x_3，y_3，z_3）とすると、

（１）加算（異なる２点）

$x_3 = AB$
$y_3 = C(A^2 x_1 z_2 - B) - A^3 y_1 z_2$
$z_3 = A^3 z_1 z_2$

但し、$A = x_2 z_1 - x_1 z_2$，　$B = C^2 z_1 z_2 - A^2 D$
$C = y_2 z_1 - y_1 z_2$，　$D = x_2 z_1 + x_1 z_2$

（２）２倍算（同じ点）

$x_3 = 2AB$
$y_3 = C(4D - B) - 8E^2$
$z_3 = 8A^3$

但し、$A = y_1 z_1$，　$B = C^2 - 8D$，　$C = 3x_1^2 + a z_1^2$，$D = x_1 E$，　$E = y_1 A$

で表される。実際には、変数ａ，ｂに対して、

乗算　$a \times b$　→　$\mathrm{REDC}(a \times b)$
加算　$a + b$　→　$a + b > p$の場合　$a + b - p$
　　それ以外　　$a + b$
減算　$a - b$　→　$a > b$の場合　$a - b$
　　それ以外　　$a - b + p$

の演算を、乗算剰余演算回路２と加減算回路３とを用いて行う。

【００３１】最後に、逆数演算回路６に$z_3 \times R$を入力して、$z_3 \times R \bmod p$のモンゴメリインバース$Z_3^{-1} = z_3^{-1} \times R^{-1} \times R \bmod p = z_3^{-1} \bmod p$を求めて、REDC　※

$\mathrm{REDC}((x_3 \times R \bmod p) \times Z_3^{-1}) = x_3 \times R \times z_3^{-1} \times R^{-1} \bmod p$

※$((x_3 \times R \bmod p) \times Z_3^{-1})$とREDC$((y_3 \times R \bmod p) \times Z_3^{-1})$とを計算する。逆数演算回路６は、モンゴメリ系での逆数演算アルゴリズム（モンゴメリインバースアルゴリズム）に基づいて演算を行う。演算結果は、

$$= x_3 \times z_3^{-1} \bmod p$$

$$REDC((y_3 \times R \bmod p) \times Z_3^{-1}) = y_3 \times R \times z_3^{-1} \times R^{-1} \bmod p$$

$$= y_3 \times z_3^{-1} \bmod p$$

となる。

【００３２】以上の演算処理により、楕円曲線上の開始点Ｐに対して、ｋＰを得る。そして実際の暗号化・復号化処理は、次のような手順で行う。図２は、送信者Ａと受信者Ｂとにおける暗号化・復号化処理の関係を示す模式図である。

【００３３】（送信者Ａ側での暗号化）乱数ｋに対して、

$kP = (x_1, y_1)$

$kQ_B = (x_2, y_2)$

を計算する。平文Ｍとx_2との排他的論理和をＸＯＲ回路７で計算し、暗号文Ｃを得る。（x_1，y_1，Ｃ）を受信者Ｂへ送る。

【００３４】（送信者Ｂ側での復号化）d_B（x_1，y_1）$= d_B kP = k d_B P = kQ_B = (x_2, y_2)$を得る。暗号文Ｃと$x_2$との排他的論理和をＸＯＲ回路７計算して、平文Ｍを得る。

【００３５】上述したような構成において、本発明の乗算剰余演算回路２は、従来の変数格納用のレジスタの代わりにワーキングメモリ（変数格納用とワーク用）を用いている。よって、従来より回路規模を小さくでき、また、変数のビット長の拡張にも対応できる。

【００３６】また、本発明の逆数演算回路６は、２つの加算器を有し、２組の変数（１組あたり２つの変数で合計４つの変数）に対して、１サイクルで同時に演算を行う。よって、演算時間の高速化を図れる。

【００３７】また、本発明のｐ′演算回路５と逆数演算回路６とにおいて、シフト演算処理を結線のつなぎ替えにて行うようにする。よって、これらの回路の規模を小さくできる。

【００３８】

【発明の実施の形態】以下、本発明をその実施の形態を示す図面を参照して具体的に説明する。図３～図７は、前述した図１における乗算剰余演算回路２，加減算回路３，R^2 modp 演算回路４，ｐ′演算回路５及び逆数演算回路６の各回路のハードウェア構成を示す図である。図３～図７を参照して、各回路について具体的に説明する。

【００３９】（乗算剰余演算回路）図３は、乗算剰余演算回路２の内部構成の一例を示す。図においてＦＦはフリップフロップを示し、64×64乗算器は64ビット×64ビットの乗算器、64加算器は64ビットの加算器をそれぞれ示す。本発明では、レジスタの代わりにワーキングメモリを利用する。

【００４０】本例の乗算剰余演算回路２では、途中の計算結果を格納するワーキングメモリの容量を小さくするために、乗数を分割して乗算剰余演算を行う方式を採用する。例えば、被乗数Ａ（192 ビット），乗数Ｂ（192 ビット）とした場合、乗数Ｂを64ビット単位に分割して部分積毎に剰余を求めていく。

【００４１】モンゴメリ系では、下位64ビットがすべて０になるようにｐの倍数を加算するために、通常の部分積剰余演算とは異なり、下位64ビットから計算していく必要がある。即ち、$A \times B_0$ の乗算を行い、その結果の値（192 ＋64ビット）の下位64ビットが０となるように、ｐの倍数を足していく。ｐの倍数は、下位64ビットの値にｐ′をかけた結果の下位64ビットである。次に、$A \times B_1$，$A \times B_2$ に同様の計算を行う。

【００４２】以下に、アルゴリズムを示す。

ｉ＝０；　ｉ＜３；　＋＋

ｉ←０，　Ｗ←０

$W \leftarrow A \times B_i + W$

$W \leftarrow W + (((W \bmod 2^{64}) \times p') \bmod 2^{64}) \times p$

$W \leftarrow W / 2^{64}$

【００４３】（加減算回路）図４は、加減算回路３の内部構成の一例を示す。図において、ＦＦはフリップフロップ、64ＦＦは64ビットのフリップフロップ、64加算器は64ビットの加算器をそれぞれ示す。

【００４４】加減算回路３において、コマンドレジスタに加算か減算かを設定する。設定した値（０または１）によって第１，第２のセレクタの制御が、下記表１のように行われる。

【００４５】

【表１】

表　　１

	第１のセレクタ	第２のセレクタ
加算の場合	b	－p
減算の場合	－b	p

【００４６】ａ，ｂ，ｐを64ビット単位で下位から演算を行い、各回の演算の結果を、下記表２のように第４，第５のワーキングメモリに格納していく。このような演算・格納処理を３回行う。

【００４７】

【表２】

9　　10

表　　2

	第４のワーキングメモリ	第５のワーキングメモリ
加算の場合	a＋b	a＋b－p
減算の場合	a－b	a－b＋p

【００４８】３回目の演算後、第１，第２の64加算器でのキャリー情報とコマンドレジスタでの設定情報とにより、第４のワーキングメモリと第５のワーキングメモリとの何れの結果を選択するかを、下記表３のように、判定器にて判定する。

【００４９】

【表３】

表　　3

	キャリー１	キャリー２	判　　定
加算の場合	1	X	第５のワーキングメモリ
	0	1	第５のワーキングメモリ
	0	0	第４のワーキングメモリ
減算の場合	1	X	第４のワーキングメモリ
	0	X	第５のワーキングメモリ

【００５０】（R^2 mod p 演算回路）図５は、R^2 mod p 演算回路４の内部構成の一例を示す。図において、FFはフリップフロップ、64加算器は64ビットの加算器をそれぞれ示す。

※【００５１】ここで、pの値を192 ビットとする。即ち、$R=2^{192}$ となる値を用い、R^2mod p 演算回路４では、2^{384} mod p を計算することになる。基本的な演算方法としては、

ステップ１　　Y←R mod p,　T←192
ステップ２　　Y←Y×２ mod p
ステップ３　　T←T－１，もし、T＝０の場合はステップ４へ
　　それ以外はステップ２へ
ステップ４　　R^2 mod p ←Y

なる計算を行うことが一般的である。ここでは、計算時間の短縮化を図るために、２ビット単位での処理を行う★ことにする。つまり、★

ステップ１　　Y←R mod p,　T←192
ステップ２　　Y←Y×４ mod p
ステップ３　　T←T－２，もし、T＝０の場合はステップ４へ
　　それ以外はステップ２へ
ステップ４　　R^2 mod p ←Y

なる計算を行う。

【００５２】このようなY×４ mod pの計算をハードウェアで行う場合、なにも引かない、－p，－２p，－３pの剰余演算が必要となり、並列に行うには、４個の加算器が必要になる。本例のR^2 mod p 演算回路４では、このような演算を表４に示すようなアルゴリズムを用いて２個の加算器（第１，第２の64加算器）で行い、回路規模の削減と計算の高速化とを実現している。つまり、被除数Yの上位３ビット，除数Pの上位２ビットに応じて、以下の表４のような処理を行う。

【００５３】

【表４】

11　　12

表　　4

Y	P	処　　理	結　果
０００	１Ｘ	剰余計算を行わずに２ビットシフト	±０
００１	１Ｘ	２ビットシフト後-p	-p
０１０	１０	１ビットシフト後-p　または　２ビットシフト後-p	-pまたは-2p
０１０	１１	２ビットシフト後-p	-p
０１１	１０	１ビットシフト後-p　さらに　１ビットシフト後-p	-2p または-3p
０１１	１１	１ビットシフト後-p　または　２ビットシフト後-p	-pまたは-2p
１ＸＸ	１Ｘ	１ビットシフト後-p　さらに　１ビットシフト後-p	-2p または-3p

【００５４】なお、1024ビットのＲＳＡ演算の場合には、$R=2^{1024}$として$R^2 \bmod p$ 演算回路４での演算を行う。

【００５５】（p′ 演算回路）図６は、p′ 演算回路５の内部構成の一例を示す。本発明では、１ビットのシフト演算処理が結線のつなぎ替えにて行われるように構成されており、回路規模の小型化を図っている。

【００５６】モンゴメリ系では、$R \times R^{-1} - p \times p' = 1$となるp′ を用いて演算を行う。p′ 演算回路５でこ＊

ステップ１　　Y←０，　T←０，p′ ←０

ステップ２　　Y mod 2＝０の場合はY←Y＋p，　p′ ←p′ ＋2^T

ステップ３　　Y←Y／２

ステップ４　　T＝63の場合は終了　　それ以外はステップ２へ

【００５８】この演算は、下位のビットからｐのべき数倍を加えていくことにより、下位64ビットがすべて１になるようなｐを求めていく演算である。

【００５９】（逆数演算回路（モンゴメリインバース演算回路））図７は、逆数演算回路６の内部構成の一例を示す。図において、ＦＦはフリップフロップ、64ＦＦは64ビットのフリップフロップ、64加算器は64ビットの加算器をそれぞれ示す。本発明では、第１の64加算器，第２の64加算器の２個の加算器を有し、後述するようなｕ，ｖの２変数に対する第１の64加算器での処理と、後述するようなｒ，ｓの２変数に対する第２の64加算器での処理とを、１サイクルで同時に行うようにして、演算時間の短縮化を図っている。

【００６０】楕円曲線上の点の加算処理の結果、得られる座標値は、

Ｘ座標　　$x_3 \times R \bmod p$

Ｙ座標　　$y_3 \times R \bmod p$

Ｚ座標　　$z_3 \times R \bmod p$

である。ここで、Ｚ座標の逆数z_3^{-1}を求めて、最終的に座標（$x_3 \times z_3^{-1}$，$y_3 \times z_3^{-1}$）を求めるのであるが、逆数演算に以下のモンゴメリインバースアルゴリズムを用いてz_3^{-1}を求める。

【００６１】［アルゴリズム］

入力：ａ，ｂ，ｎ　（ａは奇数（ａ＞ｂ＞０，ｎはａのビット数））

＊のｐ′ を求める。上記式の両辺にmod Ｒをとることにより、、以下のようになる。

$p \times p' \equiv R - 1 \pmod R$

【００５７】Ｒは２のべき乗の値をとるため、右辺は何倍してもＲ以下のビット（本暗号ハードウェア下位64ビット）に関しては、ｐ×ｐ′ の値がすべて１になるように、ｐ′ を選べば良い。ｐ′ を求めるアルゴリズムは以下のとおりである。

出力：「互いに素でない」または$b^{-1} \times 2^n \bmod a$

第１フェーズ

ｕ←ａ，　ｖ←ｂ，　ｒ←０，　ｓ←１

ｋ←０

ｖ＝０になるまで、以下を繰り返す。

ｕが偶数の場合　　ｕ←ｕ／２，　ｓ←２ｓ

ｕが奇数の場合

ｖが偶数のとき

ｖ←ｖ／２，　ｒ←２ｒ

ｖが奇数のとき

ｕ＞ｖの場合　ｕ←（ｕ－ｖ）／２，ｒ←ｒ＋ｓ，ｓ←２ｓ

ｕ≦ｖの場合　ｖ←（ｖ－ｕ）／２，ｓ←ｒ＋ｓ，ｒ←２ｒ

ｋ←ｋ＋１

ｕ≠１なら　「互いに素でない」を返す。

ｒ≧ａなら　ｒ←ｒ－ａ

第２フェーズ

ｉ＝１；　ｉ≦ｋ－ｎ；＋＋

ｒが偶数の場合　　ｒ←ｒ／２

ｒが奇数の場合　　ｒ←（ｒ＋ａ）／２

ａ－ｒを返す。

【００６２】$R=2^{192}$ よりｎ＝192(固定）となり、ｋ＜ｎの場合が出てくる。そのため、上記アルゴリズムの第２フェーズを以下のように、拡張する。

13

第２フェーズ
ｋ≧ｎの場合
ｉ＝１；　ｉ≦ｋ－ｎ；＋＋
ｒが偶数のとき　　ｒ←ｒ／２
ｒが奇数のとき　　ｒ←（ｒ＋ａ）／２
ｋ＜ｎの場合
ｉ＝１；　ｉ≦ｋ－ｎ；＋＋
ｒ←２ｒ
ｒ＞ａのとき　ｒ←ｒ－ａ
それ以外は　　そのまま
ａ－ｒを返す。

【００６３】また、逆数を計算するアルゴリズムとして、拡張ユークリッドが知られているが、モンゴメリインバースアルゴリズムでは、計算の途中に負の値をとらないため、ハードウェアでの実現に有効である。

【００６４】上述した本例の乗算剰余演算回路２，加減算回路３，$R^2 \bmod p$ 演算回路４及びｐ′演算回路５の回路規模の合計は約５万８千ゲートであり、これに上述の逆数演算回路６の約５千ゲートの回路規模の追加により、楕円暗号演算が可能となる。また、上述した本回路構成での楕円暗号演算の処理時間は、約60ｍｓｅｃ（クロック５ＭＨｚ，データ長192 ビット）であり、逆数演算回路（モンゴメリインバース演算回路）６の処理時間は、約0.6 ｍｓｅｃと評価できる。

【００６５】

14

*【発明の効果】以上説明したように、本発明によれば、乗算剰余演算回路に、加減算回路と逆数演算回路とを付加することにより、ＲＳＡ暗号演算と楕円暗号演算との両演算が可能となる。また、レジスタの代わりにワーキングメモリを使って回路規模の小型化を実現することにより、演算チップの省スペース化という効果がある。

【図面の簡単な説明】

【図１】本発明の原理説明図である。

【図２】暗号化・復号化処理の関係を示す模式図である。

【図３】乗算剰余演算回路の構成図である。

【図４】加減算回路の構成図である。

【図５】$R^2 \bmod p$ 演算回路の構成図である。

【図６】ｐ′演算回路の構成図である。

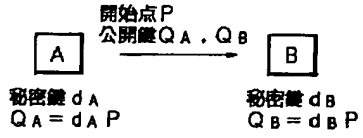
【図７】逆数演算回路（モンゴメリインバース演算回路）の構成図である。

【符号の説明】

1　演算装置
2　乗算剰余演算回路
3　加減算回路
4　$R^2 \bmod p$ 演算回路
5　ｐ′演算回路
6　逆数演算回路（モンゴメリインバース演算回路）
7　ＸＯＲ（Exclusive OR）回路

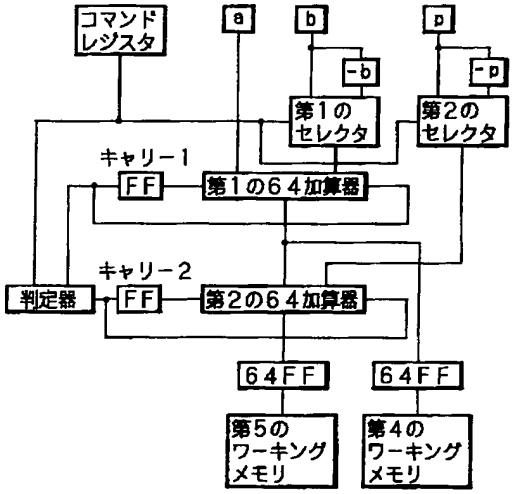
【図２】

暗号化・復号化処理の関係を示す模式図



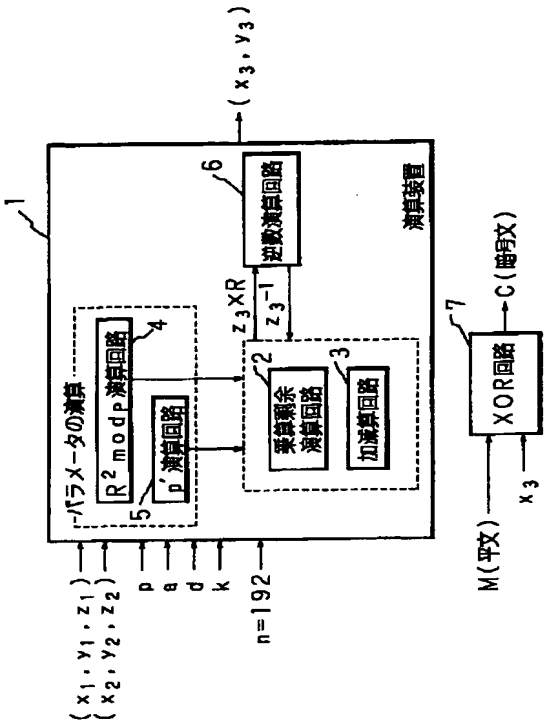
【図４】

加減算回路の構成図



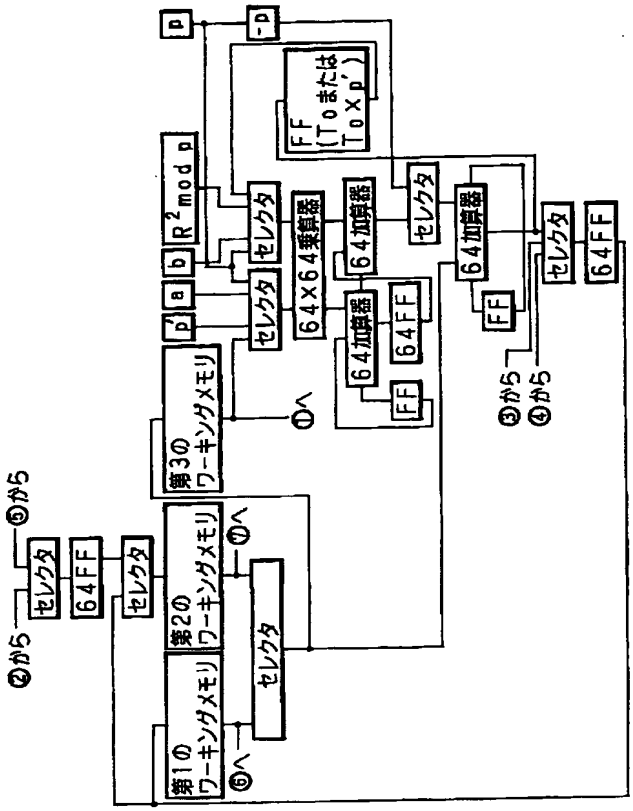
【図１】

本発明の原理説明図



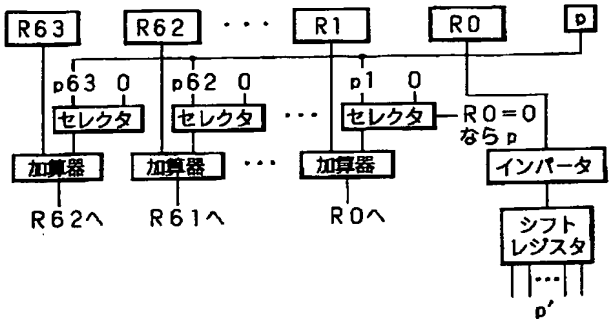
【図３】

乗算剰余演算回路の構成図

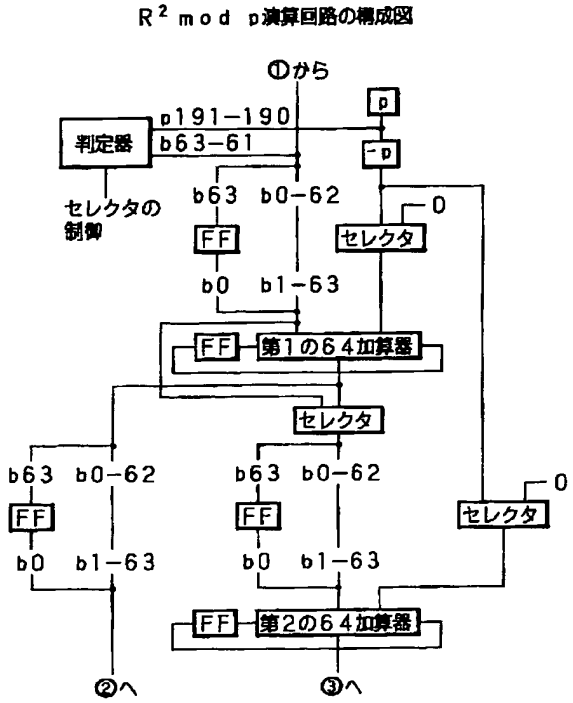


【図６】

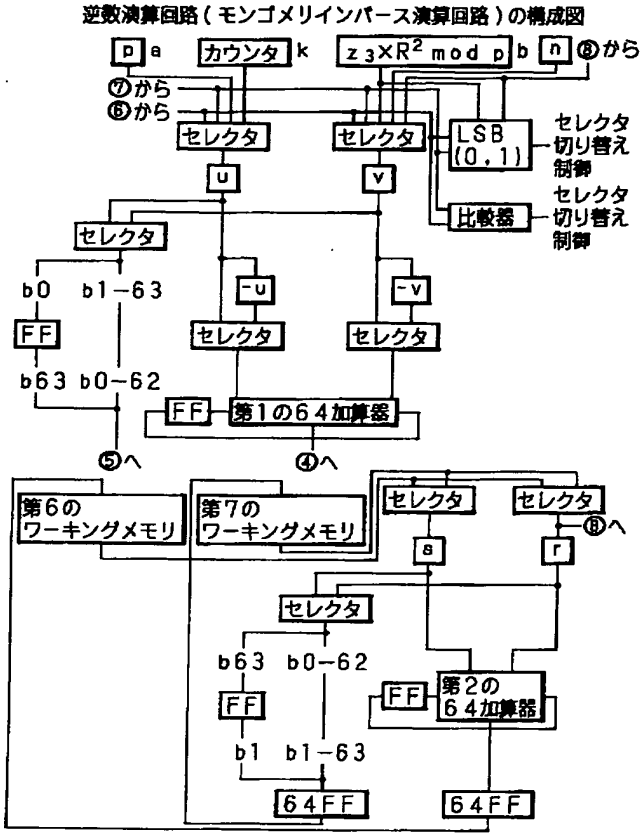
p' 演算回路の構成図



【図５】



【図７】



フロントページの続き

(72)発明者　長谷部　高行
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内

(72)発明者　武仲　正彦
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内